Pertti Järvinen

Reflection of own experiences on computing in a steel factory at early 1960s

DEPARTMENT OF COMPUTER SCIENCES
UNIVERSITY OF TAMPERE

D-2007-2

TAMPERE 2007

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER SCIENCES
SERIES OF PUBLICATIONS D – NET PUBLICATIONS
D-2007-2, MARCH 2007

Pertti Järvinen

Reflection of own experiences on computing in a steel factory at early 1960s

Accepted for presentation at HiNC2, History of Nordic Computing, Turku, August 21-23, 2007.

DEPARTMENT OF COMPUTER SCIENCES
FIN-33014 UNIVERSITY OF TAMPERE

ISBN 978-951-44-6901-5
ISSN 1795-4274

Reflection of own experiences on computing in a steel factory at early 1960s

Pertti Järvinen
Department of Computer Sciences
FIN-33014 University of Tampere, Finland
pj@cs.uta.fi

Abstract. Everything can be best seen from the historical perspective. What did the first pioneers in the information technology departments of the Finnish manufacturing companies? I had a special chance to work at early 1960s in the steel industry which had long traditions to use rather advanced tools and methods to intensify their productivity. The first computer in our company had such novel properties as movable disk packs making a direct access of stored data possible. In this paper I try describe the following issues and innovations: a) transition from the punched card machines to the computer era, b) the use of advanced programming language to intensify production of new computer software, c) trials to draw pictures by using the line printer, d) to support steel making with mathematical software, e) to store executable programs to the disk memory and to call and move them from there to the core memory for running, and f) to build a simple report generator. I shall also pay attention to the breakthrough in those innovations and in this way demonstrate how some computing solutions already then were in germ.

Introduction

Mason et al. (1997a) said that historical analyses broaden our understanding of those processes by which information technology is introduced into organizations and of the forces that shape its use. They call “dominant design”, a new configuration of an organization’s technology, strategy, and structure. A dominant design is manifested in several ways: a new organizational infrastructure, new functionality, new products, new services, new production functions, or new cost structures. By changing the basis of competition in the industry, a firm that institutes a dominant design secures an initial competitive edge. According to Mason et al. (1997a) Information Systems research literature contains very few examples of historical analyses.

In this paper I shall describe some key issues and few computing solutions to shed light on a pioneer manufacturing company and its first years to utilize a computer. According to Mason et al. (1997b) historical research offers perspectives on phenomena that are unavailable by any other methodological means. They reflect the cultural circumstances and ideological assumptions that underlie phenomena and the role played by key decision makers together with long-term economic, social, and political forces in creating them. Based on my recent efforts at collecting various research methods (Järvinen 2004), I can say that a historical method is a rarity in the literature of methodology.

The rest of the paper consists of the following topics: Introduction to the computer usage, Fortran programs for administrative purposes, visualizing some reports, supporting the making of the stainless steel by computer, towards a primitive operating system and the computer-aided development of reporting software.

On transition from the punched card machines to the computer era

My description concerns the OVAKO steel factory at Imatra in Finland. The company at 1963 bought its first computer, IBM 1401 with punched card reader, line printer, operator console and four discs units with movable disk packs. The latter were rather new. The IBM marketing men and consultants said that it was then the second newest computer with the same sort in Europe. To relate our hardware with some other installation at the same period, I refer to McKenney et al. (1997) who mentioned IBM 1401 in their famous case of Bank of America, but they described how magnetic tapes for storing bank accounts were used in that bank. The Bank of America nicely describes both the path dependency (Cohen and Levinthal 1990) and the importance of the selection decision in transitions from the earlier hardware generation to the next one.

I started at 1963 with three other IT colleagues (RH, AK, and PS). I consider those colleagues as IT experts, because they were only ones who could design and realize computer programs. My colleagues because of their economic education were mainly employed for implementing such administrative applications as payroll, invoicing, order-processing, book-keeping and budgeting. The company hired me because my result in the IBM programmer test was acceptably high, and my job concerned industrial applications, because I as a mathematician also had some knowledge about physics and chemistry. My working period started June 1, about three months before the installation of IBM 1401. First, I participated in the Fortran programming course organized by IBM.

An important observation was that the earlier punched card experts were not able to easily move to the computer time, although our computer used punched cards as input media. The stored program and especially disk memory were quite strange to punched card experts. For example, the chief of the earlier punched card department was designed a new payroll system for a computer, and his sketch of the new system was based on 17 sum-cards. The latter meant that the intermediate results in a particular phase of wage calculation process were stored to a new card (sum-card) which was thereafter punched as an intermediate output and later read as an intermediate input for the next phase of that calculation. This example demonstrates that “when novelty increases, the path-dependent nature of knowledge has negative effects because the common knowledge used in the past may not have the capacity to represent novelties now present” (Carlile 2004).

The Fortran programs for administrative purposes

In different places of the factory there were certain people (more than 30 in continuous three-shift-work) for performing production inspection (PI). Those PI people recorded every event and state-transition considered important. Based on their data different kinds of production and deviation reports were manually produced.

The new computer was evaluated to be very expensive. The local management wished that visible results could be produced as soon as possible. For programming there were two compilers available, one for an assembly language (called Autocoder) and another one for the Fortran language, mainly intended for mathematical calculations. The expressions for input and output in the Fortran language were then very restricted and simple, but the language itself was quite easy to learn. Although with Autocoder language it was possible to read all kinds of special markings punched on cards, and although in Autocoder there were especially wide range of expressions for printed output, it demanded a rather long time to become familiar with all the features of Autocoder. I therefore in the beginning of my job as a programmer selected Fortran which I used in my programming efforts. My first task was to

develop the computer programs that would produce the similar reports on production and exceptional events as earlier done manually. About one year later I changed those Fortran programs to the Autocoder programs with better output quality.

Drawing by using a line printer

In the steel making process molten steel was cast into moulds and after solidifying the ingots were moved and set down to thermal ingot furnaces for 2-4 hours before lifting them up for rolling. The number of hollows was 4 or 5. Their used capacity, percentual share in every hour per day was described as a figure which earlier manually drawn by one worker. The production inspection people recorded all the processing phases of ingots and in this way produced raw data for drawing. The figure concerning used capacity of all the ingot heating furnaces was produced once per day.

To produce the same figure with computer was not a trivial task, although there were times by the clock of ingots both when set down and when lifted up. Some ingots were not immediately put into the ingot heating furnaces but they were allowed to get totally cool, and later were taken into the ingot heating furnace. Their heating could then need many hours, and the heating period could continue from one calendar day to the next one. The ingot heating furnace history of the previous day should be re-constructed at the beginning of the development of the today's figure. The consideration of clock times required a special care in the program. The local manager, the main user of the figure, gave strong criticism based on bad appearance in the first versions of those figures.

I later saw how the Cascade project (Aanstad et al. 1971) built a graph production system. Its purpose was to produce a hardcopy version of information analysis documentation, in a proper format. Documentation consisted of tables , matrices and graphs.

To manufacture of stainless steel

The main part of steel production from the factory was carried to for different construction steels and for railway building as rails and base plates. The portion of stainless steel, although small, was increasing. Main part of stainless steel had type 18/8 or 18/10; it meant that percentage of nickel was 8 or 10 % and chromium 18 %, rather expensive raw material. The acid sustainable steel in addition contained small amount of molybdenum about 3 %, very expensive raw material. In this section I describe how I utilized computer calculations in manufacturing stainless steel.

In all the production of steel the starting point was scrap. Iron ore was used a rather minor portion. The scrap was first put into furnace and it was then melted by using electricity. From the melted batch the chemical analysis was then taken. In the factory there was built a very efficient arrangement with pneumatic mail for taking this analysis in the chemical laboratory. It took only one minute. After knowing the content of the initial batch, the suitable amounts of different additional materials, for example, Ni, FeCr, FeMo, SiCr and CaSi for slack reduction (Kostamo, 2007), was then added to the initial batch. Before adding new materials some harmful stuffs was taken away, if necessary.

In the process of making stainless steel chromium and nickel could get from the initial batch and from different additional materials, some of them containing different contents of chromium and nickel. The experts of stainless steel knew that the all the chromium and nickel insisting in additional materials moved to the final stainless steel. This fact helped the calculations how much different additional materials should be added into the initial batch. This problem could be mathematically described as equation group of 7 equations.

After discussion with the technical boss of the melting department I developed a computer program to solve that 7 equations group. In practice, after the chemical analysis of the initial batch was achieved to the furnace the foreman took a telephone call to the computer room and told the analysis. The operator then gave those analysis data into my program by using the console. It took about 30 seconds to calculate and print the result back to the console: Please, add material A m kg, material B n kg, … etc. The telephone line was kept open and after the results were ready the operator told the result to the foreman.

During the first series of stainless steel making about 25 smelting charges were made. The technical boss was at the smelting plant and I in the computer room. Manufacturing of one batch took about 4 hours to make, the first series took more than one week to make. Sometimes both the technical boss and I must wake up in the middle of night for taking care of this calculation. But I was happy, because all the made batches went inside of the very tight limits, i.e. no smelting batch was a scrap.

In steel industry Fabian (1958, 1963) rather early 1950s applied linear programming to all stages of steel making – from coal and ore through finished products. Some sub model is close to I prepared. The Fabian's largest process model covers the whole production. Bo Nyholm encouraged me to consider a similar model, but the complexity of product assortment with many production paths on the one hand and the shortage of computer memory and suitable program package on the hand prevented realization of our attempt.

Systems and application programs from the punched cards format to disk storage

The sorting program produced by IBM for our computer IBM 1401 contained 2 cases, about 4000 punched cards. It took many minutes to load it from the card reader into the core memory. When I follow the reading process I found that about the half of the cards were read at the steady rate. After more careful study I found that those cards belonged to the sub program intended for sorting data at magnetic tapes, but we did not have any tape unit. I removed those cards, and thereafter the sorting program functioned correctly in our context with four disc units.

The reduced set of the punched cards belonging to our sorting program was still rather large. Its input from the card reader even then took a long time. I therefore continued my studies to shorten the loading time. I got an idea to locate that sorting program to a disc unit. After recording it in the disc unit, the sorting program could be moved to the core memory by a short and simple call from the console. The operators were happy, because they saved time in two respects. The loading time was then shorter than before, and the loading always succeeded which was not always true with punched cards. After many repeated usage times they got worse and created a jam in the card reader.

After my first successful trial to utilize disk memory for the sorting program I applied the same idea to my application programs. I recorded them into the disk memory and they could be called by name from the operators' console. At the same time the so called IOCS (Input Output Control System) cards were eliminated from the front of the program cards. Later I understood that those IOCS cards were the beginning of an operating system, and my arrangement was in fact a simple operating system.

The next step forward was to avoid the upper limit of the core memory, 12 K. I compiled my large program as components and located every component to the disk memory. When

executing the large program I read or my main program read one component after another from the disk to the core memory. In this way I could prepare about 100 K program and execute it without any problem. Later I understood that I had applied an idea of the virtual memory and its static (preplanned) approach to storage allocation (Denning 1970).

A simple report generator

The most first tasks to be programmed were: read a set of punched cards and write a report. Later a major part of report requests concerned data in different files stored in disks. The structure of a reporting program was almost always similar. This created a wish to automate my programming efforts. Hence I developed a special program for reporting purposes. Later I recognized that I in fact developed a simple report generator.

It was possible to give the name of sorted file as a parameter for my report generator. In addition, a user must give the names of data items moved from the file into the report. The order of data items determined the presentation order in the output form. A certain output item could computed from stored data items. The expression how to perform those computations could be given by a "mathematical" formula allowing addition, subtraction, multiplication and division operations and brackets. My report program interpreted and evaluated the expression in run time, and produced an output to a certain location on the report. The general or total sums and intermediate sums could be counted. After leaving the steel factory 1967 I heard (Ruotsi 2006) that my report generator was many years used for various kinds of purposes and it functioned as a simple spreadsheet program, too.

The most demanding task in the development of the report generator was an evaluation of the mathematical expression. Later I understood that I in fact solved the problem how to transform a recursion to iterations (Kurki-Suonio 1971, p. 37).

Our report generator differed from ordinary application programs in many ways, for example, in interpretive flexibility. Doherty et al. (2006) define interpretive flexibility as the capacity of a specific technology to sustain divergent opinions. They have also found that "all technologies offer a range of functions and features that will facilitate some activities, while inhibiting others. Based upon the evidence from the empirical study, it became clear that there were upper and lower limits with respect to the functions that the system supported, and that these boundaries constrained the way in which the technology could be interpreted. More specifically, it was possible to discern, what we have termed, '*enforcing constraints*' that make certain elements of the system's functionality mandatory. At the opposite end of the spectrum, it was also possible to identify '*proscribing constraints*' that delineate the functions that do not exist, or for whatever reason cannot be used." Because our report generator was more flexible than any single report program, its interpretive flexibility was much larger than any report program, or it cannot be included into the domain of the interpretive flexibility concept at all.

My report generator was the first step in the sequence of my trials in computer-aided design of information systems. The next step in early 1980s was a simple file generator which demonstrated how it was easier to support human memory by computing systems than human data processing (Järvinen 1983). My group's last step in late 1980s was to develop an application generator, Genera (Järvinen et al. 1987). It was based on the interpreter capable to analyze and execute Pascal-type specifications. Some 20-30 applications were quickly generated with our Genera until the commercial application generators made it obsolete.

Discussion

I demonstrated that the transition from punched card machines to a computer made big changes in storing data. Computers can support people's memory with storage media allowing quick storing and retrieval properties. I also showed how the third generation programming language, even such one intended to mathematical calculations, can intensify software production compared with the traditional solution of that time, an assembly language. In order to eliminate manual work I used the computer to draw some figure. The only device for that purpose was the line printer, not very suitable for.

In addition to those primitive and easy computer applications, I also used a computer for some demanding tasks. First, to solve the set of seven equations is impossible with paper and pencil at the blast furnace with noise and heat. In this task the computer is superior compared wit a human being. We then also demonstrated networking in the germinal form. Secondly, we utilized the disk memory of our computer to improve operators' work by storing our program to disk and calling them into running from there. Our advances are clearly steps towards modern operating systems. Thirdly, I developed a report generator with spreadsheet facilities. In our construction I needed knowledge later theorized in connection with compilers.

Gaines and Shaw (1986) were a few of the first researchers who performed a historical analysis of hardware/software, state of artificial intelligence and state of human-computer interaction. They structured their analysis into eight years periods based on new generations of IBM big computers. They especially studied consecutive phases of the development of human-computer interaction. They used the model of the six eras as follows: "Each technology … seems to follow a course in which a *breakthrough* leads to successive eras: first *replications* in which the breakthrough results are copied widely; second *empiricism* in which pragmatic rules for good design are generated from experience; third *theory* in which the increasing number of pragmatic rules leads to the development of deeper principles that generate them; fourth *automation* in design based on the theory; finally leading to an era of *maturity* and mass production based on the automation and resulting in a rapid cost decline." By referring to the model of six eras I can say that my innovations or breakthroughs can be found in the computer literature, but were not available in our company. Few people (if any) in Finland then knew those innovations and their design concepts (van Aken 2004). Knowledge and algorithms concerning construction of compilers (Aho and Ullman 1972, 1973) and operating systems (Brinch Hansen 1973) were already published in the scientific literature in 1960s and early 1970s. But the March and Smith's seminal article of design research was published as late as 1995. That article outlines what is design science in Information Systems, and what are the potential results. March and Smith first wrote that in addition to new design knowledge the new instantiations also can be accepted as research outcomes. Hevner et al. (2004) later supported that claim.

References

Aanstad P., G. Skylstad and A. Sølvberg (1971), Cascade – a computer-based documentation system, In Bubenko, Langefors and Sølvberg (Eds), Computer-aided information systems analysis and design, Studentlitteratur, Lund, 93-118.

Aho A.V. and J.D. Ullman (1972), The theory of parsing, translation and compiling, Vol I: Parsing, Prentice-Hall, Englewood Cliffs.

Aho A.V. and J.D. Ullman (1973), The theory of parsing, translation and compiling, Vol II: Compiling, Prentice-Hall, Englewood Cliffs.

Brinch Hansen P. (1973), Operating system principles, Prentice Hall, Englewood Cliffs.

Carlile P. R. (2004), Transferring, translating and transforming: An integrative framework for managing knowledge across boundaries, Organization Science 15, No 5, 555-568.
Cohen W.M. and D.A. Levinthal (1990), Absorptive capacity: A new perspective on learning and innovation, Administrative Science Quarterly 35, No 1, 128-152.
Denning P. (1970), Virtual memory, Computing Surveys 2, No 3, 153-189.
Doherty N.F., C.R. Coombs and J. Loan-Clarke (2006), A re-conceptualization of the interpretive flexibility of information technologies: Redressing the balance between the social and the technical, European Journal of Information Systems 15, No 6, 569-582.
Fabian T. (1958), A linear programming model of integrated iron and steel production, Management Science 4, No 4, 415-449.
Fabian T. (1963), Process analysis of the U.S. iron and steel industry, in A.S. Manne and H.M. Markowitz (Eds), Proceedings of a Conference sponsored by the Cowles Foundation for Research in Economics at Yale University (April 24-26, 1961, Wiley, New York, 237-263. (see http://cowles.econ.yale.edu/P/cm/m18/m18-09.pdf)
Gaines B.R. and M.L.G. Shaw (1986), From timesharing to the sixth generation: the development of human-computer interaction. Part I, Int. J. Man-Machine Studies 24, No 1, 1-24.
Hevner A.R., S.T. March, J. Park and S. Ram (2004), Design science in information systems research, MIS Quarterly 28, No 1, 75-105.
Järvinen P. (1983), The ABC system – A collection of research articles, Report A112, Dept. of Mathematical Sciences, University of Tampere.
Järvinen P. (2004), On research methods, Opinpajan kirja, Tampere, Finland.
Järvinen P., P. Kiukkonen, M. Koskivirta and H. Välimäki (1987), How flexible software could support learning?, presented in Social implications of home interactive telematics (HIT) conference, June 24-27, 1987, Amsterdam, 16 p.
Kostamo P. Manager of Steel department (2007), interview 19.2.2007.
Kurki-Suonio R. (1971), Computability and formal languages, Studentlitteratur, Lund.
March S.T. and G.F. Smith (1995), Design and natural science research on information technology, Decision Support Systems 15, 251-266.
Mason R.O., J.L. McKenney and D.G. Copeland (1997a), Developing an historical tradition in MIS research, MIS Quarterly 21, No 3, 257-278.
Mason R.O., J.L. McKenney and D.G. Copeland (1997b), An historical method for MIS research: Steps and assumptions, MIS Quarterly 21, No 3, 307-320.
McKenney J.L., R.O. Mason and D.G. Copeland (1997), Bank of America: The crest and trough of technological leadership, MIS Quarterly 21, No 3, 321-353.
Ruotsi E. Manager of Production Inspection Department (2006), interview 13.11.2006.
van Aken J.E. (2004), Management research based on the paradigm of the design sciences: The quest for field-tested and grounded technological rules, Journal of Management Studies 41, No 2, 219-246.